

Decision Theory: Sequential Decisions

Computer Science cpsc322, Lecture 34

(Textbook Chpt 9.3)





3

Nov, 28, 2012

“Single” Action vs. Sequence of Actions

Set of primitive decisions that can be treated as **a single macro decision** to be made *before acting*



- 
- Agent makes observations 
 - Decides on an action 
 - Carries out the action 

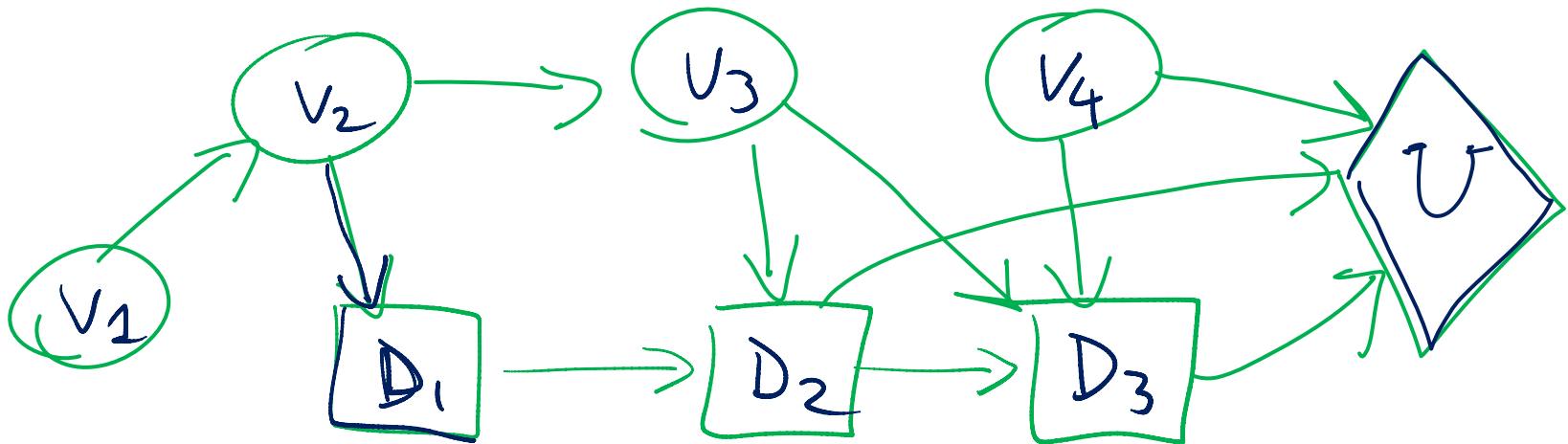
Lecture Overview

- Sequential Decisions
 - Representation ↙
 - Policies ↙
- Finding Optimal Policies ↙

Sequential decision problems

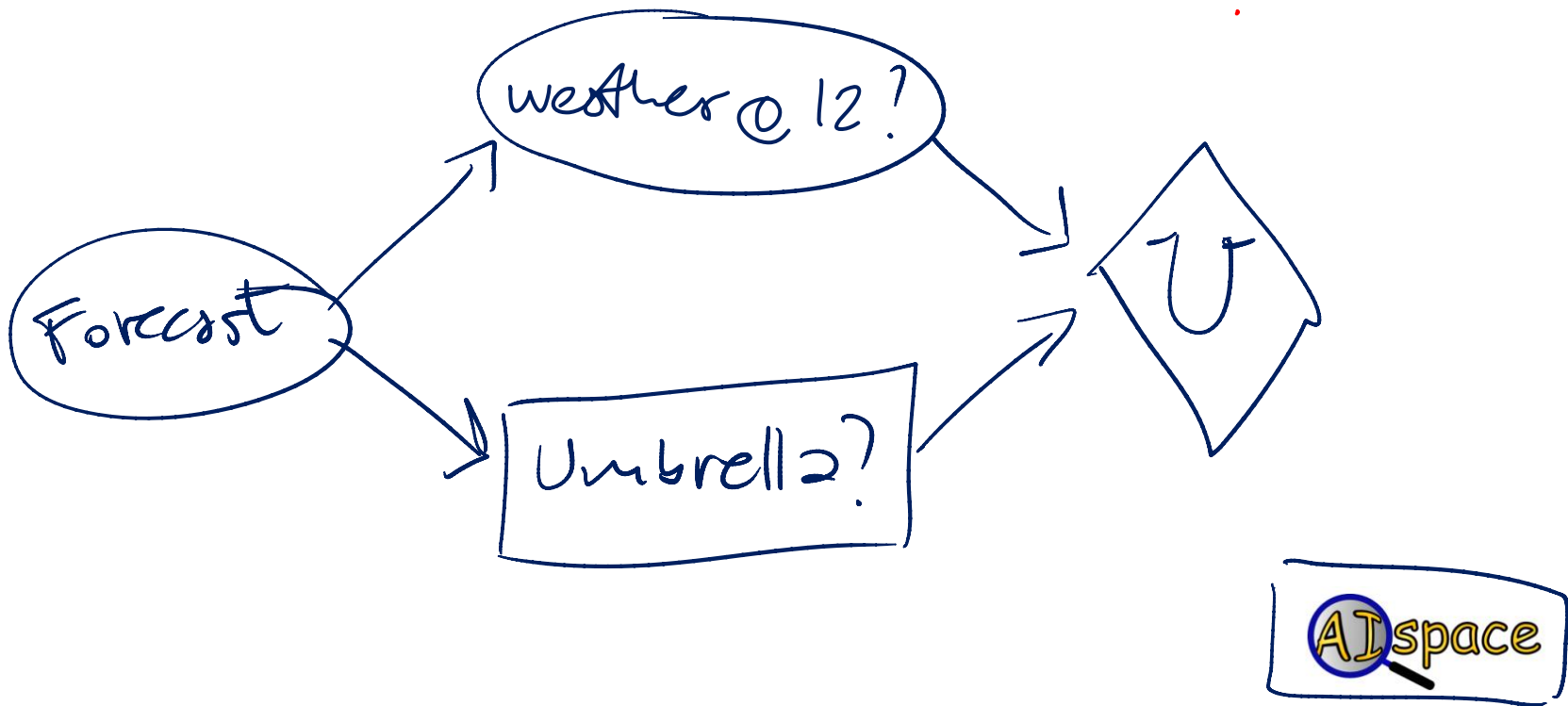
- A **sequential decision problem** consists of a sequence of decision variables D_1, \dots, D_n .
- Each D_i has an **information set** of variables pD_i , whose value will be known at the time decision D_i is made.

$$pD_3 = \{D_2, V_3, V_4\}$$



Sequential decisions : Simplest possible

- Only one decision! (but different from one-off decisions)
- Early in the morning. Shall I take my **umbrella** today? (I'll have to go for a long walk at noon)
- Relevant Random Variables?



Policies for Sequential Decision Problem: Intro

- A **policy** specifies what an agent should do under each circumstance (for each decision, consider the parents of the decision node)

In the *Umbrella* “degenerate” case:

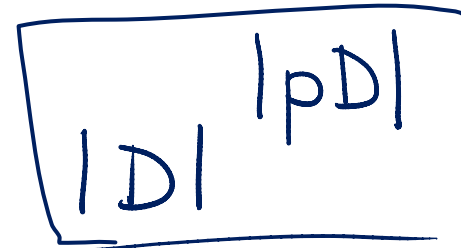
D_1 ? T F

pD_1 Rainy
 Cloudy
 Sunny

How many policies? 2^3

One possible Policy

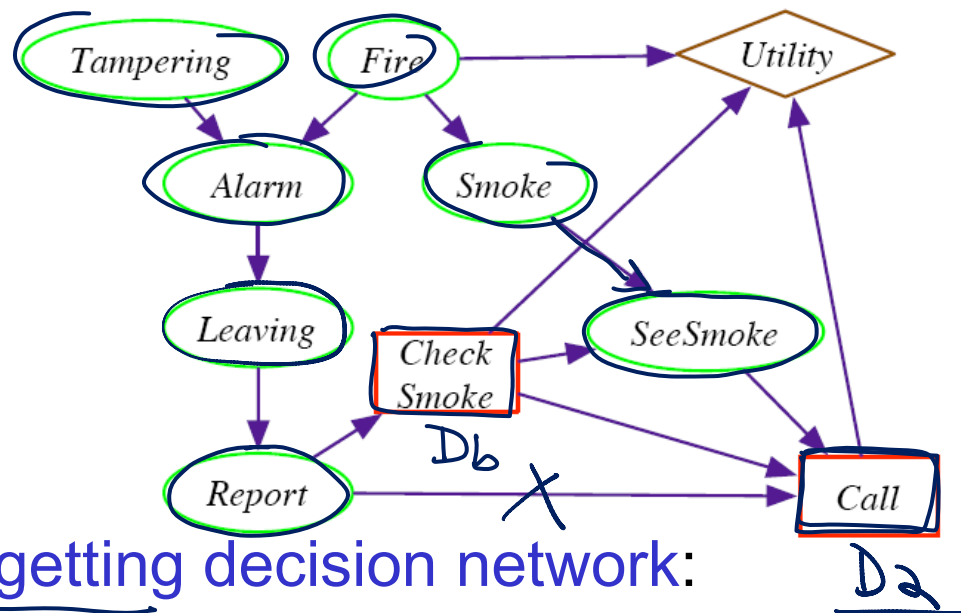
→ R T F T...
→ C T F T...
→ S F F T...



3 policies

Sequential decision problems: “complete” Example

- A **sequential decision problem** consists of a sequence of decision variables D_1, \dots, D_n .
- Each D_i has an **information set** of variables pD_i , whose value will be known at the time decision D_i is made.



$$pCS = \{R\}$$
$$pC = \{R, CS, SS\}$$

No-forgetting decision network:

- decisions are totally ordered
- if a decision D_b comes before D_a , then
 - D_b is a parent of D_a
 - any parent of D_b is a parent of D_a


$$pCS \subseteq pC$$

Policies for Sequential Decision Problems

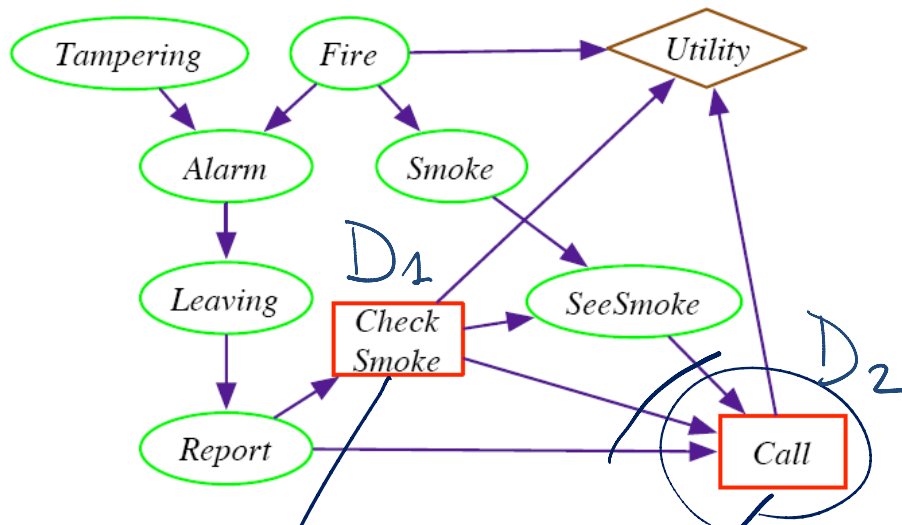
- A **policy** is a sequence of $\delta_1, \dots, \delta_n$ **decision functions**

$$\delta_i : \underline{\text{dom}(pD_i)} \rightarrow \underline{\text{dom}(D_i)}$$

- This policy means that when the agent has observed $O \in \text{dom}(pD_i)$, it will do $\delta_i(O)$

Example: δ_1

Report	Check Smoke	
T	T T F	F
F	T F T	F



How many policies?

$2^2 * 2^8$

δ_2

Report	CheckSmoke	SeeSmoke	Call
true	true	true	true
true	true	false	false
true	false	true	true
true	false	false	false
false	true	true	true
false	true	false	false
false	false	true	false
false	false	false	false

Lecture Overview

- Recap
- Sequential Decisions
- **Finding Optimal Policies**

When does a possible world satisfy a policy?

- A possible world specifies a value for each random variable and each decision variable.
- Possible world w satisfies policy δ** , written $w \models \delta$ if the value of each decision variable is the value selected by its decision function in the policy (when applied in w).

$w \neq \delta$

VARs

VARs	
Fire	true
Tampering	false
Alarm	true
Leaving	true
Report	false
Smoke	true
SeeSmoke	true
CheckSmoke	true
Call	true

Rand.

Dec.

Decision function for...

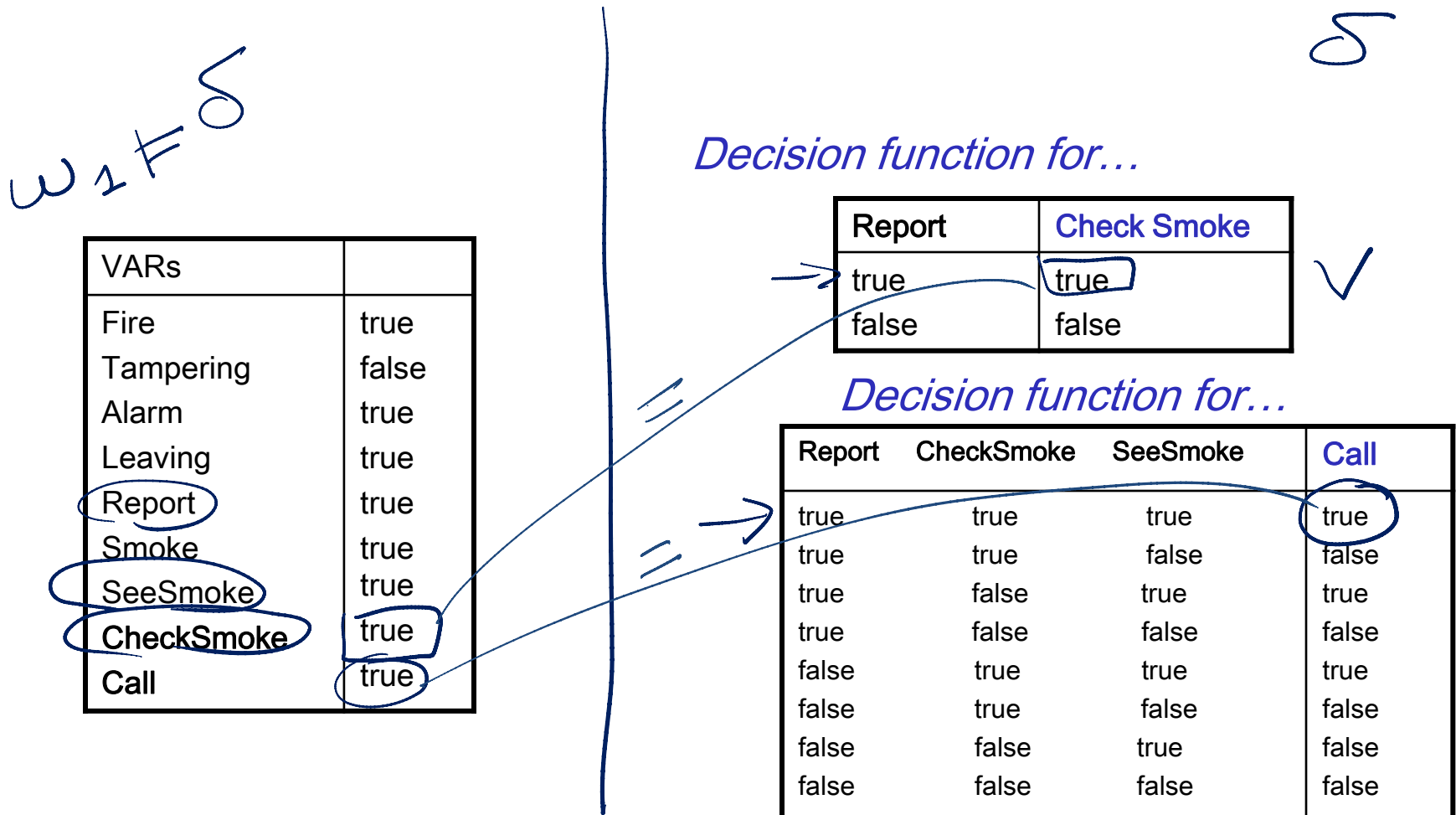
Report	Check Smoke
true	true
false	false

Decision function for...

Report	CheckSmoke	SeeSmoke	Call
true	true	true	true
true	true	false	false
true	false	true	true
true	false	false	false
false	true	true	true
false	true	false	false
false	false	true	false
false	false	false	false

When does a possible world satisfy a policy?

- Possible world w satisfies policy δ , written $w \models \delta$ if the value of each decision variable is the value selected by its decision function in the policy (when applied in w).



Expected Value of a Policy

- Each possible world w has a probability $P(w)$ and a utility $U(w)$
- The expected utility of policy δ is

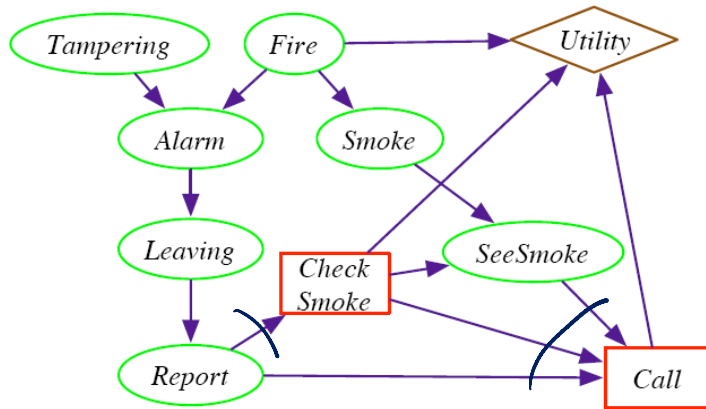
$$\sum_{w \models \delta} P(w) \cdot U(w)$$

- The optimal policy is one with the max expected utility.

Lecture Overview

- Recap
- Sequential Decisions
- Finding Optimal Policies (Efficiently)

Complexity of finding the optimal policy: how many policies?



- How many assignments to parents?
 $C \leq 2 \quad C \leq 2^3$
- How many decision functions? (binary decisions)
 $2^2 \quad 2^3$
- How many policies? *product*
 $2^2 * 2^3$

- If a decision D has k binary parents, how many assignments of values to the parents are there?

$$2^k$$

- If there are b possible actions (possible values for D), how many different decision functions are there?

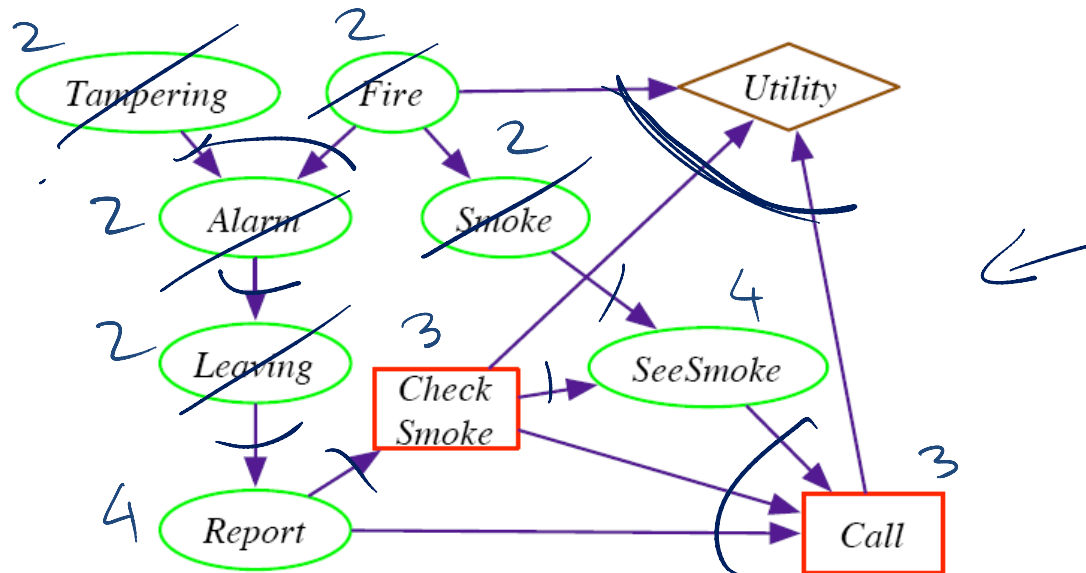
$$b^{2^k}$$

- If there are d decisions, each with k binary parents and b possible actions, how many policies are there?

$$(b^{2^k})^d$$

Finding the optimal policy more efficiently: VE

1. Create a factor for each conditional probability table and a factor for the utility.
2. Sum out random variables that are not parents of a decision node.
3. Eliminate (aka sum out) the **decision variables**
4. Sum out the remaining **random variables**.
5. Multiply the factors: this is the **expected utility of the optimal policy**.



Eliminate the decision Variables: step3 details

- Select a variable D that corresponds to the latest decision to be made
 - this variable will appear in only one factor with its parents
- Eliminate D by **maximizing**. This returns:
 - A **new factor** to use in VE, $\max_D f$
 - The **optimal decision** function for D , $\arg \max_D f$
- Repeat till there are no more decision nodes.

Example: Eliminate CheckSmoke

Report	CheckSmoke	Value
true	true	-5.0
true	false	-5.6
false	true	-23.7
false	false	-17.5

Report	Value
true	-5.0
false	-17.5

New factor

Decision Function

Report	CheckSmoke
true	true
false	false

VE elimination reduces complexity of finding the optimal policy

- We have seen that, if a decision D has k binary parents, there are b possible actions, If there are d decisions,
- Then there are: $(b^{2^k})^d$ policies
- Doing variable elimination lets us find the optimal policy after considering only $d \cdot b^{2^k}$ policies (we eliminate one decision at a time)
 - VE is much more efficient than searching through policy space.
 - However, this complexity is still doubly-exponential we'll only be able to handle relatively small problems.

+ give up nonforgetting assumption
+ approx. algorithms
422

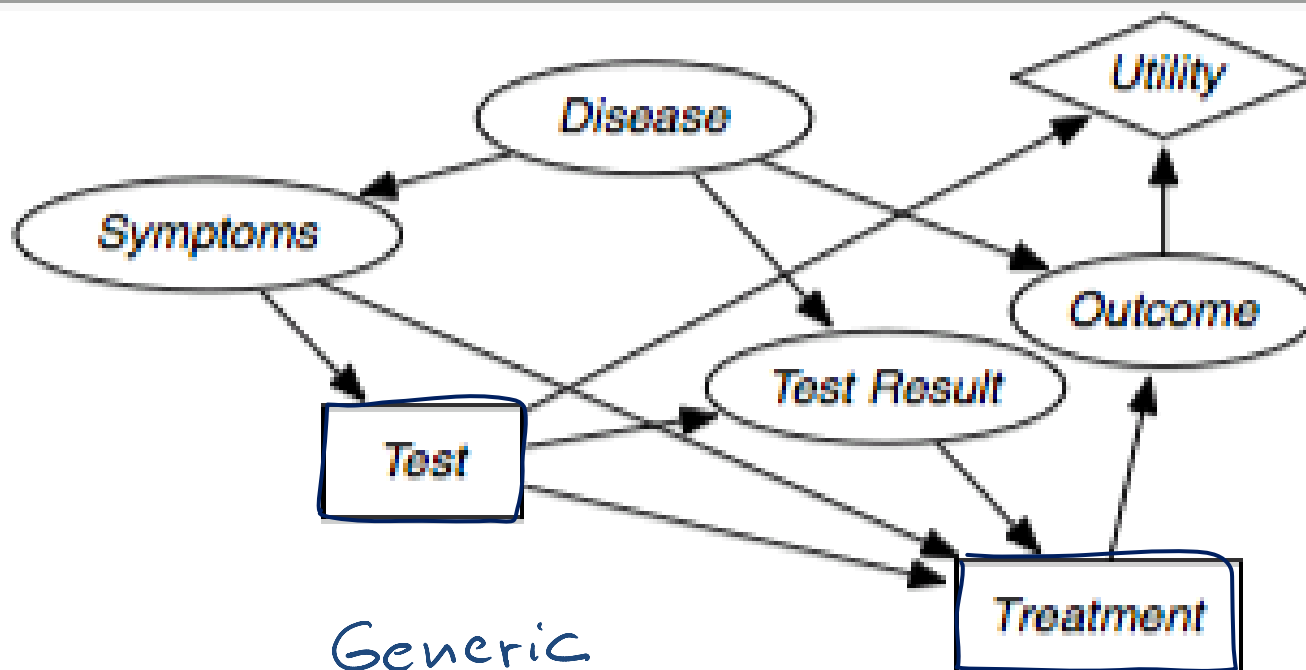


Figure 9.8: Decision network for diagnosis

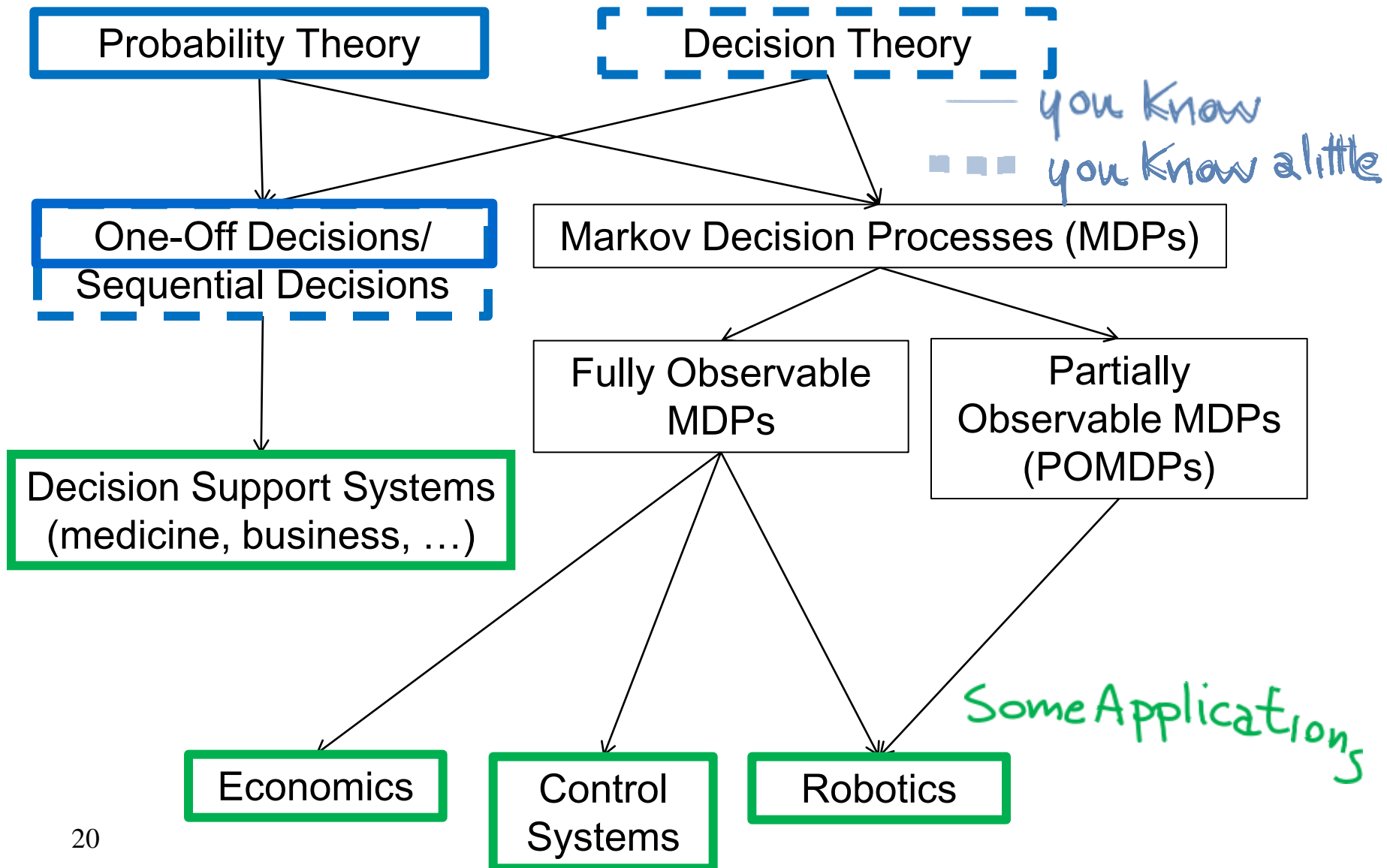
to select what test to apply
and then what treatment to prescribe

Learning Goals for today's class

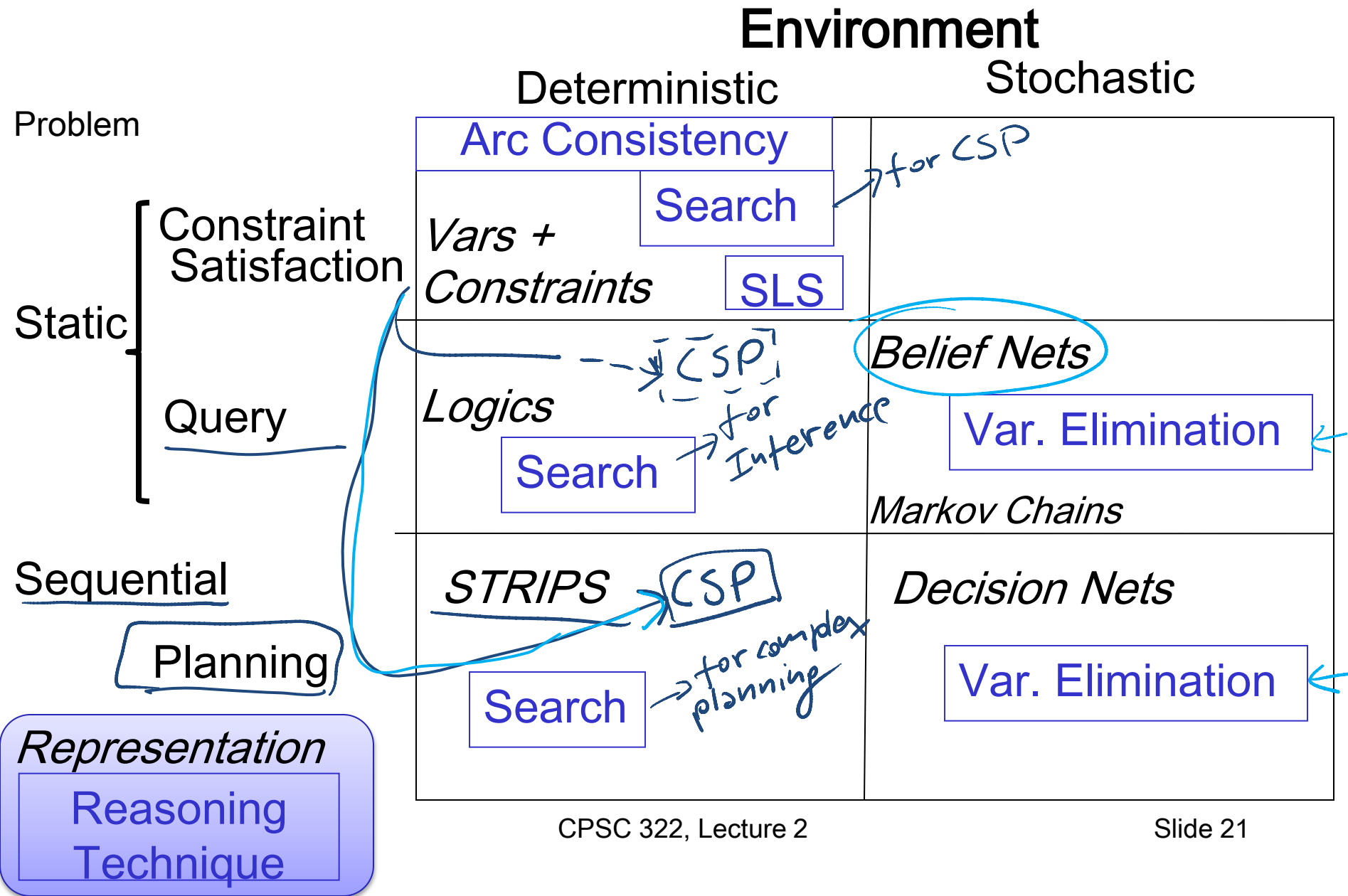
You can:

- Represent **sequential decision problems** as decision networks. And explain the **non forgetting property**
- Verify whether a possible world satisfies a policy and define the expected value of a policy
- Compute the number of policies for a decision problem
- Compute the **optimal policy** by Variable Elimination

Big Picture: Planning under Uncertainty



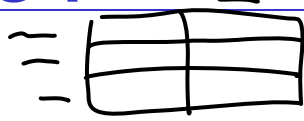
Cpsc 322 Big Picture



After 322

349

322 big picture



Deterministic

Stochastic

- Machine Learning
- Knowledge Acquisition
- Preference Elicitation

CSPs

Vars + Constraints

Techniques to study
SLS Performance

Logics

- First Order Logics
- Temporal reasoning
- Description Logics

Belief Nets

More sophisticated
reasoning

Markov Chains and HMMs

*Hierarchical Task
Networks*

Partial Order Planning

*Markov Decision Processes
and*

Partially Observable MDP

More sophisticated
reasoning

Where are the
components of our
representations
coming from?

The probabilities?

The utilities?

The logical formulas?

From people and
from data!

Applications of AI

Announcements

Homework #4, due date: Fri Nov 30, 1PM.

You can drop it at my office (ICICS 105) or by handin.

- **FINAL EXAM:** Thu Dec13, 3:30 pm (3 hours, PHRM 1101)

Final will comprise: ^{6-8 pts} 10 -15 short questions + ^{~20 pts} 3-4 problems

- Work on all practice exercises (including 9.B) and sample problems
- While you revise the learning goals, work on review questions
 - I may even reuse some verbatim 😊
- Come to remaining Office hours!

- ~~Fill out~~ **Online Teaching Evaluations Survey.**